

Selling your Software as a Service

Kyle Cordes
http://kylecordes.com
kyle@kylecordes.com

St. Louis Code Camp 2007
http://stlcodecamp.org
May 5, 2007

An Encouraging Idea

When you open up an IDE/editor at home, on your own time, you have a remarkably power in your hands: you can create something that you own (modulo some employment agreements). It is yours to sell, to give away, or to decline to do either, on your own terms. Likewise a company has this on a larger scale.

For the last few years my firm has been selling access to an application we wrote (and continue to write) as a service, charging a monthly fee for use, hosting, and support. I won't name names or describe our product in detail (this talk is not spam), but I will share some things we've learned along the way.

What is SAAS?

Delivering a software product (especially a complex "enterprise" system, less so with consumer products) involves more than just bits in a file / on a CD-ROM:

- 1) Delivery of the deployable code
- 2) License (legal right to use it)
- 3) Hosting: hardware, deployment, and management thereof
- 4) Support for the server(s) / software on the servers
- 5) Support for the end users / software on their PCs or other devices

In a consulting or in-house project, the most common type of project for the audience here at Code Camp, you sell hours of labor to do the things above.

In a commercial software product company, you bundle #1 and #2 for a price, then perhaps sell a service contract for #4 and possibly #5.

With Software as a Service (SAAS) you bundle #1, #2, #3, #4, and possibly #5, then sell the result for a yearly / monthly fee times some measure of volume (per computer, per user, per transaction, per site, etc.). A company selling their software in this way is sometimes called an Application Service Provider.

The Business Model

From a business perspective, SAAS offers a recurring revenue stream rather than "spiky" up front sales. In the short term this can make higher investment demands (since the revenue arrives more slowly). In the long term, though, recurring revenues are very amenable to a sustainable ongoing business.



On the downside, as a SAAS vendor you are at greater risk of ongoing payments stopping, vs. up-front payments in full when the sale is made.

Collections are typically much less of a problem with SAAS: you will generally be paid by customers in a timely way (as they would pay for electricity, bandwidth, and other ongoing utility services).

This gradual arrive of revenues will shape your company's financial decisions: you need to meter expenses to match.

Hosting

With SAAS, your customer stays out of the "hosting business", but you get in to it. One way to offer reliable hosting is to put your servers in a colo facility; you can typically get a full rack (with power and bandwidth) for less than \$1000 per month. This will provide redundancy in the power, air conditioning, and connectivity, which would be much more costly to provide in-house.

You will become experts in hosting, by experience. You will host more instances of your software than any one customer would, so you are in a better position to do a good job doing so.

Another route, for a smaller operation, is to rent a single dedicated server or even a "virtual" dedicated server. Or, go utterly virtual and use

Amazon EC2 and S3 as your infrastructure.

Support / Operations

As an SAAS firm you can offer much more hands-on support than a traditional software vendor, because you have direct and immediate access to the production systems serving your customers.

At the same time, this access will expose your team to the customer's pain points, making it more likely those problems are fixed soon.

With SAAS you can readily deploy updates regularly (weekly / monthly) at low cost, enable end-to-end agility. Paul Graham wrote about this in "The Other Road Ahead".

What do customers like about SAAS?

Customer **love** not paying for unproven software up front, even if they have abundant cash and even if the software is in use elsewhere. They avoid making an unrecoverable investment in case their needs change or the software turns out to be inadequate.

Customers generally like not running a hosting operation: not operating a data center, not hiring employees for it, etc.

Customers benefit with SAAS's easy scaling up or down as their needs grow and shrink. Financial decision makers often prefer costs that scale smoothly with business volume.

What do customers not like about SAAS?

Customer may be concerned that they have less direct control over an application supplied SAAS, than one hosted in-house, with less ability to persuade an operator/developer to drop everything and fix it now.

Eventually a customer may spend more with SAAS (at least in terms of the checks written to the software vendor, though perhaps not in total cost) than they would have spent buying a license up-front; thus they are prone to regret that they didn't buy an up front license.

A customer may want to customize the software in a way the vendor does not offer.

Start Fast

SAAS provides considerable flexibility in delaying software work past the product launch; "admin" features can be less complete and more rough, if they are used primarily by the vendor's staff instead of by the customer, and can require deeper system knowledge.

There is also a risk that the ability to get by with rough edges will too easily allow you to never truly finish the product.

What's it like to sell software this way?

The biggest difference in selling SAAS is a more substantial long term relationship with the customer, both operationally and financially. This can be a great advantage – with a close customer relationship you will be in a position to understand their needs more fully, and innovate in your software to meet those needs better.

Customizations

The purest SAAS way to offer customizations is to not charge hourly / project fees for the changes, but rather to invest in the desired enhancements then charge a higher ongoing price for the enhanced system. Hybrid customization payments are also possible, combinations of customization fees and monthly costs. Be sure your customer understands that paying for a customization, does not grant ownership of the underlying system.

Intellectual Property

While an SAAS firm usually owns (or licenses) its software, customers should own their data (and have access to a copy of the raw data, if they ask for it). Hire a lawyer to work out the details.

Go to great lengths to keep customer data separate: you must make sure to never, even briefly or accidentally, give one customer access to another customer's data. For small, cheap services, WHERE clauses are OK for this, but for large expensive services, use (at minimum) separate databases and server code instances, and perhaps even separate hardware.